

# CyberTalents Challenges

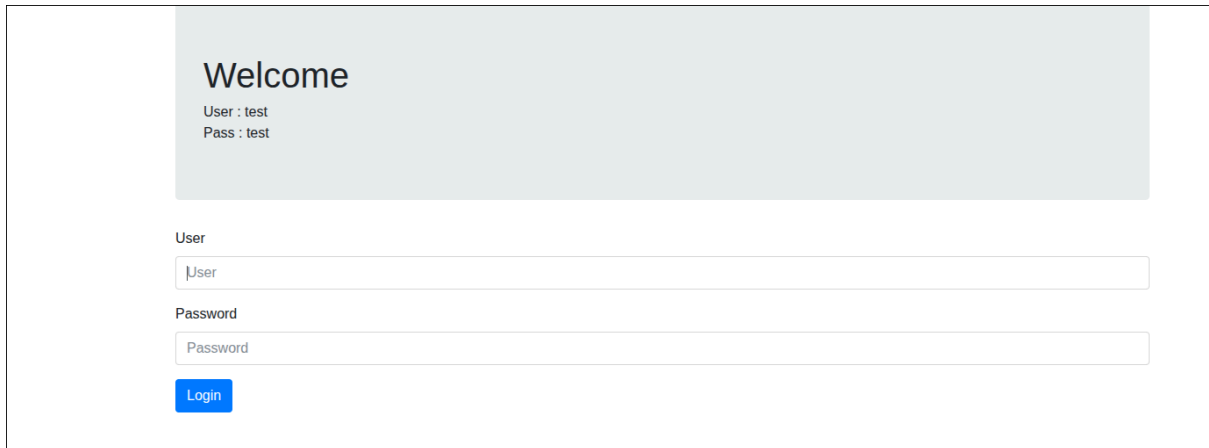
Web Security – admin gate first – level medium

Author: levith4n

Description:

Flag is safe in the admin account info

Ini adalah tampilan utama dari aplikasi web target, kita dapat melihat kita diberikan kredensial untuk melakukan pengujian.



Welcome

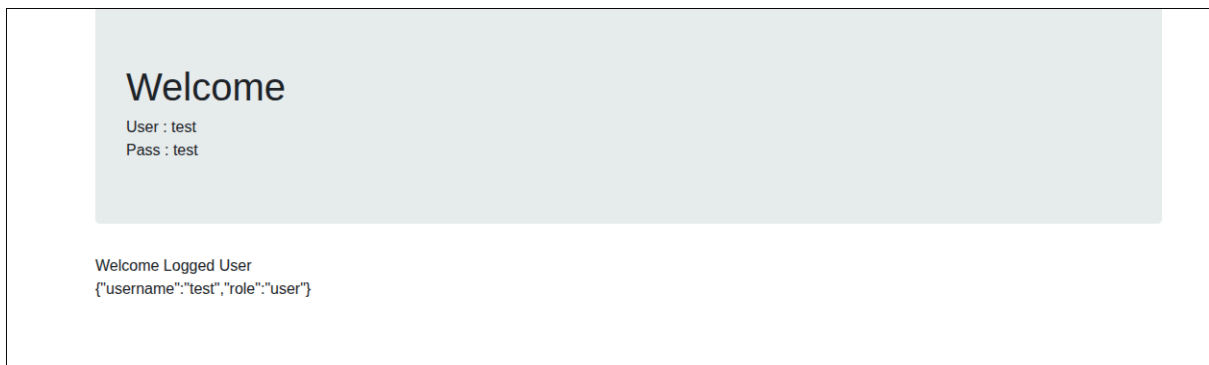
User : test  
Pass : test

User

Password

Login

Setelah login, kita diberikan sebuah informasi penting, yaitu *role* kita adalah **user**.



Welcome

User : test  
Pass : test

Welcome Logged User  
{ "username": "test", "role": "user" }

Kemudian, saya mencoba membaca *source code html* dan menemukan bahwa kode di bawah digunakan untuk mengambil nilai *cookie* dan menggunakannya sebagai *Bearer token* untuk autentikasi pada *header* permintaan AJAX.

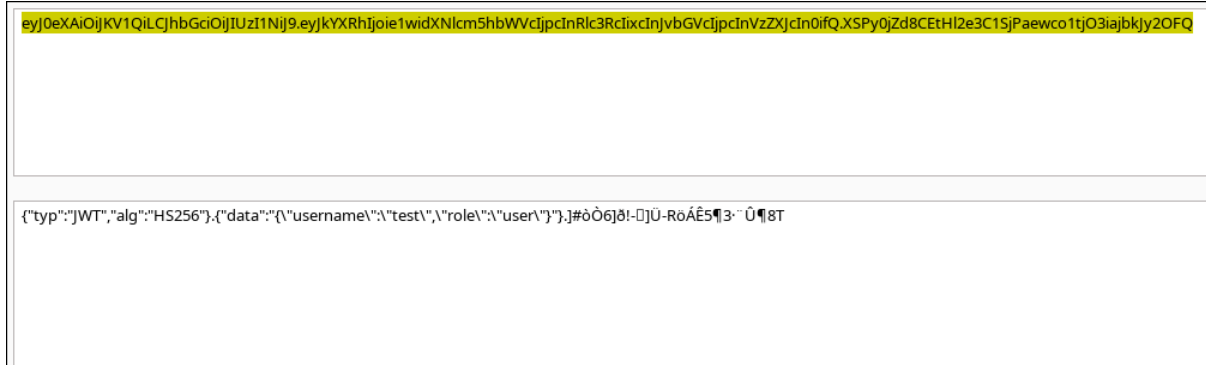
```
<div>Welcome Logged User</div>
<div id="info"></div>
<script type="text/javascript">
    function getCookie(name) {
        var value = "; " + document.cookie;
        var parts = value.split("; " + name + "=");
        if (parts.length == 2) return parts.pop().split(";").shift();
    }

    var token = getCookie('auth');
    function getMyInfo()
    {
        console.log('checking logged user info');
        $.ajax({
            url: 'index.php?info=yes',
            type: 'GET',
            dataType: 'json',
            contentType: "application/json",
            beforeSend: function(xhr) {
                xhr.setRequestHeader("Authorization", "Bearer "+token)
            },
            success: function(data){
                $('#info').html(JSON.stringify(data));
            },
            error: function(x,y,z){
                console.log(x);
                console.log(y);
                console.log(z);
            }
        });
    }
    getMyInfo();
</script>
</div>
<!-- Optional JavaScript -->
<!-- jQuery first, then Popper.js, then Bootstrap JS -->
```

Jika kita lihat menggunakan *Burp intercept*, maka terdapat header *Authorization* yang nilainya sama dengan nilai pada *cookie*. Melihat strukturnya, saya curiga ini adalah **JWT**, karena biasanya **JWT** itu memiliki 3 bagian yang dipisahkan oleh titik (*header.payload.signature*).



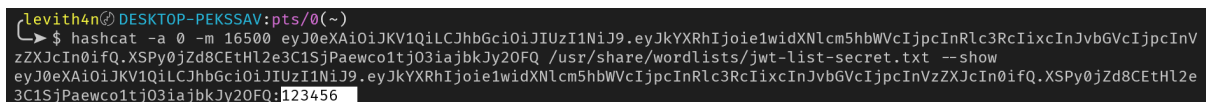
Berdasarkan strukturnya yang khas (**tiga bagian ter-encode Base64**), saya memvalidasi token ini dengan **Burp Decoder**. Hasil *decode* pada segmen pertama (header) mengkonfirmasi bahwa ini adalah **JWT** yang ditandatangani dengan algoritma **HS256**.



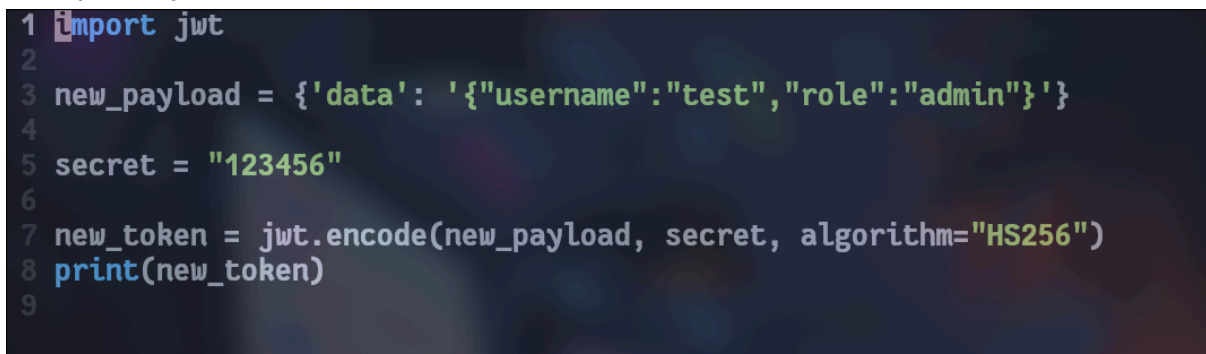
Selanjutnya, saya mencoba melakukan **JWT Secret Key Cracking** menggunakan **hashcat**:

- **-a** digunakan untuk menentukan mode serangan diikuti dengan sebuah angka 0, angka 0 ini menandakan bahwa kita menggunakan *dictionary attack*.
- **-m** adalah metode hash yang kita serang, angka 16500 menunjukkan JWT.

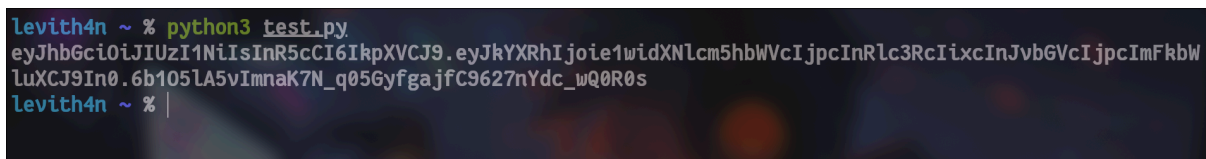
Seperti yang terlihat, kita berhasil menemukan *secret key*-nya yaitu **123456**.



Setelah mendapatkan *secret key* yang digunakan untuk menandatangani *header* dan *payload* pada token tersebut, sekarang saya akan menggunakan *secret key* ini untuk menandatangani dan *payload* kustom yang saya buat, yaitu untuk user *test* saya ubah *role*-nya menjadi **admin**.



Setelah dijalankan, token jwt kustom sudah siap kita gunakan.



Kita akan menempa token yang sudah kita siapkan pada **Bearer**, seperti yang terlihat kita berhasil mendapatkan flag.

Request					Response				
Pretty	Raw	Hex			Pretty	Raw	Hex	Render	
<pre>1 GET /index.php?info=yes HTTP/1.1 2 Host: wcamxw132pue3se6m51xvkoecl304m73p5l12bq3-web.cyberntalentslabs.com 3 Authorization: Bearer   eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJkYXRhIjoieWVudCmlucyBjbG9ja3Rlcixicm9udGVudCI6ImFkbWw1UXVjZDQ0Lm91dG8tLS1hbnNpdG9ka3Rlcixicm9udGVudCiwqd0R0S 4 X-Requested-With: XMLHttpRequest 5 Accept-Language: en-US,en;q=0.9 6 Accept: application/json, text/javascript; */*; q=0.01 7 Content-Type: application/json 8 User-Agent: Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko)   Chrome/137.0.0 Safari/537.36 9 Referer: http://wcamxw132pue3se6m51xvkoecl304m73p5l12bq3-web.cyberntalentslabs.com/ 10 Accept-Encoding: gzip, deflate, br 11 Cookie: auth=   eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJkYXRhIjoieWVudCmlucyBjbG9ja3Rlcixicm9udGVudCiwqd0R0S.XSPy8jZd8CEtH2e3c1SJPaewcoitJO3iajbKjyZOZFQ 12 Connection: keep-alive 13 14</pre>					<pre>1 HTTP/1.1 200 OK 2 Server: nginx/1.27.1 3 Date: Tue, 01 Jul 2025 07:30:41 GMT 4 Content-Type: application/json;charset=utf-8 5 Connection: keep-alive 6 Content-Length: 61 7 8 {   "username": "test",   "role": "admin",   "flag": "J1WI753cr3TQ2018" }</pre>				